sdmay18-18: Fleet monitoring system
Week 5 Report
October 2 - October 16

Team Members
Tyler Hartsock — *Web Manager*
Anthony Guss — *Technical Lead*
William Fuhrmann — *Test Engineer*
Kendall Berner — *Project Manager*
Matthew Fuhrmann — *Report Manager*
Venecia Alvarez — *Point of Contact*

Summary of Progress this Report
The team worked on the first version of the design document, and on improving our website with better pictures and short biographies for the team members.

The front-end team worked on integrating the prototypes they had created earlier with the server API the server prototype that the server provides. The Google Maps API and Charts.js prototypes now pull data from the server and progress has started on recreating these pages using Angular.js.

The server team worked on improving the model from the server prototype to prepare it to accept data from the Android micro controller. This involved creating a model for a login system for fleet managers, and the associations of vehicles and drivers in a fleet to fleet managers. They evaluated the api-npm module for Node.js to determine whether it was worth integrating into the project, and decided that we do not need the features it provides and it is better to not include it in the server.

The micro controller team worked on investigating the hardware connection needed to get the provided library to connect to the ODB-II port of a vehicle, which was not provided to us. They worked with our client, Lotfi, to order the cable that will allow this connection. They finished the prototype that shows the location services results for the application, but had trouble deploying it to the board. They created a new prototype that sends dummy data to the server using its data acceptance API and the new manager model. They investigated the possible use of the openxc-android library (https://github.com/openxc/openxc-android/tree/master/library/src), which was ultimately unsuited to our project due to hardware differences. They attended two research meetings with people working on automotive programming applications.

Pending Issues
We are waiting for the cable needed to connect the micro controller to the ODB-II port to arrive. We are having trouble deploying the Android application for location services testing to the hardware.

Plans for Upcoming Reporting Period
The whole team plans to begin looking at the feedback made on the Project Plan v1 and see what we need to improve for v2.

The front-end team plans to work on integrating their prototype webpages with Angular.js, while making new pages to display new information stored in the server model.

The server team plans to transition into working on the micro controller, as the server is in a good state at the moment. They will begin to research how the CAN BUS network works and might start to work on the CAN BUS interfacing library based on the example code we have.

The micro controller team plans to determine whether the location services works on the device, and to learn more about the CAN BUS network and the example code so that they can start working on creating a library that our app can use to interface with the vehicles.

Individual Contributions

Team Member	Contribution	Weekly Hours	Total Hours
Tyler Hartsock	Tyler has been having some family issues that have prevented him from being able to work on the project. These issues should be resolved soon, and he should be able to get back to work soon.	0	10
Anthony Guss	Worked on the models for the server making them more reflective of what our final models will look like. This involved adding a data array to the current model that will hold the data passed from the Android device. I also created a manager model that includes the base of a login system for managers for the website, as well as a way to assign managers vehicles and get what vehicles a manager owns. This will allow the website to pull data for specific vehicles that a manager owns. I also spent time researching api-npm, a module for Node. I was looking into its viability and whether or not we should use it for our server. After spending time working on it, I decided that its use was outside of the scope of the project, as many of the features it added are not needed. Adding it at this point would add unnecessary complications to our project. I also worked on adding information pertaining to the server aspect of our design document.	7.5	20.75
William Fuhrmann	Created a prototype Android application that builds a JSON object from a set of dummy data, a data class that will hold data queried from CAN BUS (once that works), and sends data to the server via POST api. The prototype allows the driver to input his driver ID which then posts that info to the server. Then it will put vehicle data to the server	7	23.25

	periodically. Also worked on adding micro-controller and Android parts to the project plan and the design document.		
Kendall Berner	Got server set up and managed to call api from the client side successfully. Used data received to create custom chart on the spot with Chart.js. Spent some time upgrading our team website as well. Reworked the Meet the Team page to use a carousel. Also refactored lots of the website code to be more managable in the future. Changed the pages to php so I could include a header rather than have huge amounts of repeated code in every html file.	8	26
Matthew Fuhrmann	I researched the connection that needs to be made between our micro controller and the CAN BUS network, as no OBD-II cables were provided and there was little to no documentation of how they are supposed to be connected. My client assisted me and we worked together with the board manufacturer to order a cable that should allow us to start to connect to the CAN BUS network. I confirmed that the micro controller has sufficient Wi-Fi connectivity, and attempted to find if it has sufficiently accurate location services with and without GPS, but came across some Android problems (trouble deploying my first Android app). This is very close to finished: we have it working on other Android devices and emulators, all we need to is to figure out the problem in putting it on the specific micro controller we are using. I evaluated the possible use of a library for the CAN BUS interfacing on Android (https://github.com/openxc/openxc-android/tree/master/library/src). I found that the library will not be able to be used because of how it is implemented to work with two separate hardware devices while we only have one device, and it would be too difficult to modify the library to our hardware. I also looked at the VI (Vehicle Interface)(https://github.com/openxc/vi-firmware) library source code to get a better understanding of the different operations our micro controller will have to do, including	11.5	32

	how to set the byte fields and how to work with CAN BUS's data consistency protocols. I also did some work on the design document regarding the micro controller design and the initial set-up of the document with content duplicated from the project plan.		
Venecia Alvarez	I set up my environment to use MongoDb and our Node.js server. I successfully made a call to the server through the client side. I implemented Google Maps API to display a map with markers of vehicles in a fleet. I created a dropdown menu of the vehicles in the fleet where the manager can select a specific vehicle and the map will zoom on the one selected. I also spent time converting all of the existing features I have onto a more AngularJS page. I created a timeline for our project plane, worked on the design document, and submitted my bio for the team webpage as well.	10.5	23.5